Oppgave 1A.8: En forenklet kode for stjernedannelse

D. Vader
Institute of Theoretical Astrophysics, University of Oslo,
P.O. Box 1029 Blindern, 0315 Oslo, Galactic Empire[*]
(Dated: 20. august 2019)

Vi skriver en kode som beregner $\Delta\vec{r}$ og $\Delta\vec{v}$ til partiklene ved å bruke Euler-Cromer. Fra disse så oppdaterer vi $\vec{r}$ og $\vec{v}$ i tidssteg på $\Delta t = 10^{-5}$ år. Ved å lage løkker over alle partiklene så kan vi summe opp avstand fra sentrum og finne midlere avstand r fra sentrum. I den samme løkken så finner vi midlere kinetisk energi til partiklene som vi setter lik det analytiske uttrykket $3/2kT$ for å beregne temperaturen.

I. INTRODUKSJON

Vi skal skrive en kode der vi oppdaterer $\vec{r}$ og $\vec{v}$ i tidssteg $\Delta t = 10^{-5}$ år. I hvert tidssteg så må vi beregne $\Delta\vec{r}$ og $\Delta\vec{v}$ fra gravitasjonskrafta $\vec{F}_g$. Vi indekserer alle partiklene med et skallnummer fra 0 til 99 og beregner avstanden til partiklene fra sentrum. Vi summer så over alle partiklene i skallet under for å finne M som inngår i uttrykket for $\vec{F}_g$.

I hvert av de 7×10^4 tidsstegene, så skal vi beregne midlere r over alle partikler samt midlere kinetisk energi og lage en graf over r og T som funksjon av tiden. Til slutt lager vi histogrammer av alle v_i for å se om disse enda følger Maxwell-Boltzmann.

II. METODE

Vi begynner med å lese inn dataene i arrayer. Vi lager en array `r` med dimensjon `(N,3)` med posisjonene og en array `v` også med dimensjon `(N,3)` med de innleste hastighetene. Deretter lager vi en array `shell` med dimensjon `N` som inneholder skallnummer fra 0 til 99 for hver partikkel. Denne finner vi ved å først finne avstanden til hver partikkel fra sentrum:

```
avst=sqrt{r(:,0)**2+r(:,1)**2+r(:,2)**2}
```

og deretter dele avstanden med tykkelsen til hvert skall. Ved å ta heltall av dette så får vi ut skallnummeret.

Vi lager så en annen array `npart_shell` med dimensjon `100` som gir oss antall partikler i et gitt skall. Vi lager da en løkke over alle partikler og legger til 1 i det skallet som partikkelen tilhører.

Vi lager nå en funksjon `rad` som beregner den midlere avstand til sentrum. Merk at vi ikke skal telle med partikler som har kommet langt vekk. Dette løser vi ved å sortere arrayen `avst`:

```
sort_avst=np.sort(avst)
```

Vi tar da midlet over de 80% første elementene i denne sorterte arrayen.

Vi lager så en annen funksjon `temp` som beregner temperaturen. Inne i løkken over partikler så summer vi opp

```
E+ = 0.5*m*v[i]**2
```

Til slutt finner vi temperatur:

```
T=E/(3/2*k)
```

Når vi har disse funksjonene klare så begynner vi løkka over tidssteg. Inne i løkka over tidssteg så har vi løkke over partiklene. For hvert tidssteg og hver partikkel så skal vi beregne Δr og Δv som vi trenger til Euler-Cromer. Vi beregner først Δv. Da bruker vi uttrykket for kraften

$$\vec{F}_i = -G\frac{M(r_i)m}{r^3}\vec{r}_i$$

Der m er massen til partikkelen som er gitt ved M_{tot}/N og $M(r_i)$ er totalmassen på innsiden av avstanden r_i fra sentrum. Her summer vi nå opp all partiklene i arrayen `npart_shell` for alle skall som er på innsiden av `shell(i)`. Hvis vi kombinerer dette med Newtons 2.lov $\vec{F}_i = m\vec{a}_i$ der akselrasjon jo er

$$\vec{a}_i = \frac{d\vec{v}}{dt}$$

så finner jeg Δv ved å skrive

```
dv=-g*mr[i]*m/avst[i]**3*r[i,:]*dt
```

Vi ma også få med friksjonsleddet som gir bidrag til Δv:

```
dv+=-k*rho*sqrt(v[i,0]**2+v[i,1]**2+v[i,2]**2)*v[i,:]
```

der vi beregner `rho=nshell[shell[i]]/n`. Til slutt oppdaterer vi $\vec{v}$

```
v[i,:]=v[i,:]+dv
```

Vi kan videre finne Δr ved å bruke at

$$\vec{v} = \frac{d\vec{r}}{dt}$$

som gir `dr`v*dt=. Merk at siden vi kjører Euler-Cromer, så må vi bruke den nye oppdaterte hastigheten. Dermed finner vi ny `r` ved

```
r[i,:]=r[i,:]+dr
```

Merk at vi inne i løkka over tidssteg kaller funksjonene `rad` og `temp` og lagrer verdiene vi får ut.

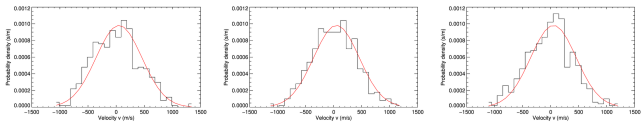
[*] dvader@astro.uio.galemp

III. RESULTATER

Ved å bruke funksjonen `temp` før det første tidssteget så finner vi $T = 20$K. Deretter bruker vi `np.histogram` for å plotte histogrammet til v_i i x, y og z-retning som vist i figur 1. Vi plotter Maxwell-Boltzmann

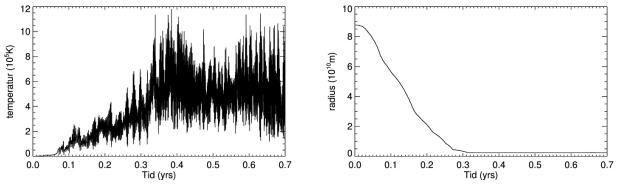
```
sigma=sqrt(k*t/m)
p[vx]=(sqrt(2*pi)*sigma)**(-1)*exp(-0.5*vx**2/sigma**2)
```

over histogrammet i hver av dimensjonene (koden her viser for v_x). Denne stemmer godt overens med histogrammet.



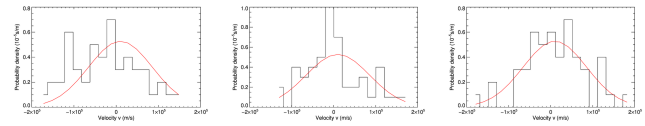
Figur 1. Histogram av hastighetskomponentene i x, y og z-retning

For å gjøre beregningene raskere, så velger vi å kjøre simuleringen med kun 50 partikler. Vi kjører så simuleringen i 70.000 tidssteg. I figur 2 ser vi hvordan midlere radius og temperatur (beregnet med funksjonene `rad` og `temp`) varierer med tiden og til slutt stabiliserer seg på en radius 0.1×10^{10}m. Vi finner at slutt-temperaturen er 541.000K som er for lavt til at fusjonsreaksjoner kan sette igang.



Figur 2. Temperatur og radius til skya som funksjon av tida

I figur 3 så viser vi histogrammet til hastighetsfordelingene til partiklene etter simuleringen samt et plott av Maxwell-Boltzmann-fordelingen for temperaturen som partiklene har i siste tidssteg. Igjen så er stemmer denne godt overens med histogrammet.



Figur 3. Histogram av hastighetskomponentene i x, y og z-retning.

Radien stabiliserer seg etter 0.35 år. Vi ser at verdien til radien er nær tykkelsen på det innerste skallet, noe som gjør at radien ikke kan bli mindre.

Til slutt kjører vi koden på nytt uten friksjonsledd. Da blir Δv alltid akselrasjon uten bremsing og partiklene stopper dermed ikke opp i sentrum.

IV. DISKUSJON

Lengden på tidssteget i Eulermetoden er det viktig å undersøke om er lite nok, da for store steg kan gi gale resultater. Er tidssteget for stort så vil bevegelsene være veldig unøyaktige. Vi kjører derfor simuleringen på nytt med 1/10 så store tidssteg, og 10 ganger så mange tidssteg totalt. Resultatene var svært konsistente, noe som bekrefter at tidssteget vårt var lite nok.

Antall skall som vi deler skya opp i vil også kunne påvirke resultatet. Jo flere skall, jo mer nøyaktig bør resultatene bli (da tykkelsen på skallet blir mindre). Vi doblet antall skall og fant at resultatene endret seg betydelig. Vi konkluderer at koden bør kjøres med tynnere skall for å få mer eksakte resultater.

V. KONKLUSJON

Vi simulerer bevegelsen til partikler ved å oppdatere $\vec{r}$ og $\vec{v}$ fra uttrykk for gravitasjons- og friksjonskraft. Tidsstegne våre på 10^{-5}år viser seg å være små nok til å gi nøyaktige resultater, men oppdelingen av skya i 100 skall er for grov og bør forbedres.

ACKNOWLEDGMENTS

Takk til R. Rutherford, W. Pauli, L. de Broglie og G. Marconi for nyttige diskusjoner under arbeidet med denne artikkelen. Takker også A. Turing og N. H. Abel for gode tips til programmeringen.

[1] Hansen, F. K., 2017, Forelesningsnotat 1A i kurset AST2000